3 years ago by Moty Michaely — 6 min read

How to connect to SFTP in Python

Learn how to connect to SFTP, list files, upload and download using Python

#Guides #Engineering

With the purpose of secure file and data transfer, SFTP has been a top choice for many users. While it is an efficient and user friendly protocol, engaging with an SFTP server from a programming language such as Python may take a tad more deciphering. With the guidance of the following post, we hope you will come out the other side feeling familiar and ready to utilize and connect to SFTP from Python.

Requirements

Prior to jumping in, some setup is required. We'll use the 'pysftp' package to connect and pass commands to the SFTP server. When you are ready to install it, manually run:

```
pip install pysftp
```

Or create a requirements.txt file and declare your dependencies within. Then, save the following as requirements.txt:

```
pysftp>=0.2.9
```

Then run:

```
pip install -r requirements.txt
```

Connecting to SFTP

In this post, we'll be using an environment variable named `SFTPTOGO_URL` that contains all the information required to connect to an SFTP server using a URI format: sftp://user:password@host. The variable is parsed to extract the URI parts using `urlparse`, and the remote server's host key is verified by default using `~/.ssh/known_hosts`.

Once the connection is established, the SFTP client object will be assigned to the variable: `connection`.

```
import pysftp
from urllib.parse import urlparse
import os


class Sftp:
    def __init__(self, hostname, username, password, port=22):
        """Constructor Method"""
        # Set connection object to None (initial value)
        self.connection = None
        self.hostname = hostname
        self.username = username
        self.password = password
        self.port = port

    def connect(self):
        """Connects to the sftp server and returns the sftp connection object"""

        try:
            # Get the sftp connection object
            self.connection = pysftp.Connection(
                host=self.hostname,
                username=self.username,
                password=self.password,
                port=self.port,
            )
        except Exception as err:
            raise Exception(err)
        finally:
            print(f"Connected to {self.hostname} as {self.username}.")


    def disconnect(self):
        """Closes the sftp connection"""
        self.connection.close()
        print(f"Disconnected from host {self.hostname}")
```

Listing Files

Now that the connection is all set up, we can use it to list files on the remote SFTP server. This is achieved by calling the connection object's `listdir` function or the `listdir_attr` function. Either one of these functions can take a remote path argument or, if omitted, will list the files and directories in the current remote directory. The `listdir` function returns a list of filenames (as string), while the `listdir_attr` function returns a list of SFTPAttributes objects containing the file size, creation, modification timestamps and permissions, in addition to the file name. In our example, our wrapper function returns a generator, allowing the function caller to iterate over the returned files.

```
    def listdir(self, remote_path):
        """lists all the files and directories in the specified path and returns
        for obj in self.connection.listdir(remote_path):
            yield obj

    def listdir_attr(self, remote_path):
        """lists all the files and directories (with their attributes) in the spe
        for attr in self.connection.listdir_attr(remote_path):
            yield attr
```

Upload File

The next step is to upload a file. Use the connection object's `put` function and pass the path to the local file and the remote path, which is also where the file should end up by the end of the upload. The function call would look like: `connection.put("./local.txt", "./remote.txt")`

```
    def upload(self, source_local_path, remote_path):
        """
        Uploads the source files from local to the sftp server.
        """

        try:
            print(
                f"uploading to {self.hostname} as {self.username} [(remote path:
            )

            # Download file from SFTP
            self.connection.put(source_local_path, remote_path)
            print("upload completed")

        except Exception as err:
            raise Exception(err)
```

Download File

The last job left for us to complete is downloading our files. Use the connection object's `get` function, and pass the path to the remote file and the local path, where you would store the downloaded file. You would call the function like so: `connection.get("./remote.txt", "./download.txt")`

```
    def download(self, remote_path, target_local_path):
        """
        Downloads the file from remote sftp server to local.
        Also, by default extracts the file to the specified target_local_path
        """

        try:
            print(
                f"downloading from {self.hostname} as {self.username} [(remote pa
            )

            # Create the target directory if it does not exist
            path, _ = os.path.split(target_local_path)
            if not os.path.isdir(path):
                try:
                    os.makedirs(path)
                except Exception as err:
                    raise Exception(err)

            # Download from remote sftp server to local
            self.connection.get(remote_path, target_local_path)
            print("download completed")

        except Exception as err:
            raise Exception(err)
```

The Whole Thing

So we've made it to the finish line! If you would like to run the entire program from start to finish, copy the following code and save it as `main.py`:

```
import pysftp
from urllib.parse import urlparse
import os


class Sftp:
    def __init__(self, hostname, username, password, port=22):
        """Constructor Method"""
        # Set connection object to None (initial value)
        self.connection = None
        self.hostname = hostname
        self.username = username
        self.password = password
        self.port = port

    def connect(self):
        """Connects to the sftp server and returns the sftp connection object"""

        try:
            # Get the sftp connection object
            self.connection = pysftp.Connection(
                host=self.hostname,
                username=self.username,
                password=self.password,
                port=self.port,
            )
        except Exception as err:
            raise Exception(err)
        finally:
            print(f"Connected to {self.hostname} as {self.username}.")

    def disconnect(self):
        """Closes the sftp connection"""
        self.connection.close()
        print(f"Disconnected from host {self.hostname}")

    def listdir(self, remote_path):
        """lists all the files and directories in the specified path and returns
        for obj in self.connection.listdir(remote_path):
            yield obj

    def listdir_attr(self, remote_path):
        """lists all the files and directories (with their attributes) in the spe
        for attr in self.connection.listdir_attr(remote_path):
            yield attr

    def download(self, remote_path, target_local_path):
        """
        Downloads the file from remote sftp server to local.
        Also, by default extracts the file to the specified target_local_path
        """

        try:
            print(
                f"downloading from {self.hostname} as {self.username} [(remote pa
            )

            # Create the target directory if it does not exist
            path, _ = os.path.split(target_local_path)
            if not os.path.isdir(path):
                try:
                    os.makedirs(path)
                except Exception as err:
                    raise Exception(err)

            # Download from remote sftp server to local
            self.connection.get(remote_path, target_local_path)
            print("download completed")

        except Exception as err:
            raise Exception(err)

    def upload(self, source_local_path, remote_path):
        """
        Uploads the source files from local to the sftp server.
        """

        try:
            print(
                f"uploading to {self.hostname} as {self.username} [(remote path:
            )

            # Download file from SFTP
            self.connection.put(source_local_path, remote_path)
            print("upload completed")

        except Exception as err:
            raise Exception(err)


if __name__ == "__main__":
    sftp_url = os.environ.get("SFTPTOGO_URL")

    if not sftp_url:
        print("First, please set environment variable SFTPTOGO_URL and try again.
        exit(0)

    parsed_url = urlparse(sftp_url)

    sftp = Sftp(
        hostname=parsed_url.hostname,
        username=parsed_url.username,
        password=parsed_url.password,
    )

    # Connect to SFTP
    sftp.connect()

    # Lists files with attributes of SFTP
    path = "/"
    print(f"List of files with attributes at location {path}:")
    for file in sftp.listdir_attr(path):
        print(file.filename, file.st_mode, file.st_size, file.st_atime, file.st_m

    # Upload files to SFTP location from local
    local_path = "/Users/saggi/Downloads/tls2.png"
    remote_path = "/tls2.png"
    sftp.upload(local_path, remote_path)

    # Lists files of SFTP location after upload
    print(f"List of files at location {path}:")
    print([f for f in sftp.listdir(path)])

    # Download files from SFTP
    sftp.download(
        remote_path, os.path.join(remote_path, local_path + '.backup')
    )

    # Disconnect from SFTP
    sftp.disconnect()
```

Finally, run it using the command:

`python main.py`

Congratulations on connecting to SFTP using Python!

Check out more code samples on Github.

You might also like

Data Subscription & SFTP Automation: Squarespace, Make.com & SFTP To Go
Laura-ann Burgess

SFTP Workday Integration—Why & How To Do It
Laura-ann Burgess

Supply Chain Automation: Data Automation w. DocParser
Laura-ann Burgess

Our 6 Top EDI Providers of 2024
Laura-ann Burgess

Newer post
How to connect to SFTP in Ruby on Rails

Older post
What is SFTP?